



开篇词 | 这样入门Go，才能少走弯路

2021-10-13 Tony Bai

《Tony Bai · Go语言第一课》

[课程介绍 >](#)



讲述：Tony Bai

时长 14:43 大小 13.48M



你好，我是白明（英文名：Tony Bai），欢迎你和我一起学习 Go 语言。

我现在在一家初创企业东软睿驰工作，是一名车联网平台的架构师，同时我也是技术博客 tonybai.com 的博主、GopherChina 大会讲师。

从 2011 年开始我便关注了 Go 语言，是 Go 语言在国内的早期接纳者。那个时候，离 Go 开源还不过两年，没有人想到它会成长到今天这样，成为后端开发的主流语言之一。

在对 Go 长达十年的跟随和研究中，我沉淀了很多个人的经验和思考。我也希望通过这节课，跟你分享我学习和使用 Go 语言的一些心法。



我与 Go 的这十年

2011 年，一次偶然的机会，我非常幸运地看到了 Go 语言之父 Rob Pike 的 Go 语言课程 [幻灯片](#)。当时我正经受着 C 语言内存管理、线程调度和跨平台运行等问题的折磨，看到 Go 语言的语法清新简洁，还支持内存垃圾回收、原生支持并发，便一见钟情。

我是个对编程语言非常“挑剔”的人，这跟我从事的方向有关。十多年来，我一直在电信领域从事高并发、高性能、大容量的网关类平台服务端的开发，这两年也进入了智能网联汽车行业。由于长期从事后端服务开发，我涉猎过很多后端编程语言。

我曾深入研究过 C++，短暂研究过 Java、Ruby、Erlang、Haskell 与 Common Lisp，但都因为复杂度、耗资源、性能不够、不适用于大规模生产等种种原因放弃了。

而且，如果你对我所在的行业有所了解，你可能会知道，我参与开发的系统对性能十分敏感。我也曾长期使用 C 语言作为生产语言，同时使用 Python 开发各种辅助工具。但是，C 语言的生产效率不高，各种陷阱也很多，而 Python 开发效率确实很高，但性能又不好。

难道就没有一门相对“完美”、符合我使用需求的编程语言吗？这个时候，Go 来了。

但在我开始接触和学习 Go 的 2011 年，Go 语言还未发布 Go 1.0 稳定版本，还处于“周发布（weekly release）”的状态。我还记得我接触的第一个 Go 版本还是 [release.r60](#)，也就比 Rob Pike 的 Go 课程里使用的版本稍新一些。

但这个时候的 Go 版本，存在着很多不尽如人意的地方，尤其是 GC 延迟比较大，成为了 Go 语言上生产环境的最大障碍。

虽然 Go 早期版本无法上生产环境，但我却一直紧跟 Go 语言的演化进程。

从 [Go 1.4 版本](#) 开始，每当 Go 发布一个大版本，我都撰文分析这个大版本中 Go 语言的主要变化。这一系列文章至今仍在继续，未来也将持续进行下去。

特别是在 Go 1.5 版本实现自举、Go 1.11 版本解决 Go 包依赖问题后，Go 语言逐渐成熟，我也逐渐尝试在生产中使用 Go。从开始替代 Python 编写一些辅助工具，到编写一些网关所需的网络协议，我发现 Go 都可以完美适用。

一直到近些年，Go 替代了 C、Python，成为了我的第一生产语言。我开始直接使用 Go 编写生产系统，诸如短信网关、5G 消息网关、MQTT 网关，还有 API 网关等等。事实证明，Go 语言不仅生产效率高，其应用的执行效率也完全能满足要求。

从 2019 年开始，我将自己每天阅读到的 Go 社区的优秀技术资料整理成公开的 [Gopher 日报](#) 供大家参阅。在国内 Go 社区中，Gopher 日报得到了圈里许多同学的欢迎。

紧跟 Go 演进十年的我，已经将 Go 语言的点点滴滴深深地烙印在大脑中。

推荐你入坑 Go 的三大理由

如果说十年前的我是因为“一见钟情”瞬间入坑 Go，那么在十年后的今天，我们应该做的是系统地、认真地思考一下为什么要选择学习 Go。

我想了想，我会从这三个角度建议你现在开始学习 Go 语言。

第一个理由：对初学者足够友善，能够快速上手。

十多年来，业界都公认：**Go 是一种非常简单的语言**。到底有多简单呢？在 2011 年，我从一个 C 语言程序员的身份开始学习 Go，使用 Rob Pike 的 Go 教程，我在一天之内就学完了 Go 的全部语法，一周内就可以编写一些简单、实用，而且质量不低的小程序了。

而且，跟现在很多逐渐添加各种特性的语言相比，Go 不仅一开始简单，直到现在也都保持“简单”。Go 的设计者们在发布 Go 1.0 版本和兼容性规范后，似乎就把主要精力放在精心打磨 Go 的实现、改进语言周边工具链，还有提升 Go 开发者体验上了。演化了十多年，Go 新增的语言特性也同样是“屈指可数”。

正因为如此，**作为静态编程语言的 Go 已经将入门门槛，降低到和动态语言一个水平线上了。**

第二个理由：生产力与性能的最佳结合。

Go 的简单和对初学者的友善可以让更多的开发者走进 Go 语言大门，但要让更多开发者留在 Go 语言世界，Go 还需体现出自己的核心竞争力。这个核心竞争力就是，**Go 语言是生产力与性能的最佳结合。**

如果你熟悉的是静态语言，那你刚好就是 Go 最初的目标用户。Go 创建的最初目的，就是构建流行的、高性能的服务器端编程语言，用以替代当时在这个领域使用较多的 Java 和 C++。而且，Go 也实现了它的这个目标。

Go 语言的性能在带有 GC 和运行时的语言中名列前茅，与不带 GC 的静态编程语言（比如 C/C++）之间也没有数量级的差距。在各大基准测试网站上，在相同的资源消耗水平的前提下，Go 的性能虽然低于 C++，但高出 Java 不少。

如果你熟悉的是动态语言，那也完全不用担心。Go 的大部分早期采用者，就来自动态语言程序员群体，包括 Python、JavaScript、Ruby 和 PHP 等语言的使用群体。因为和动态语言相比，Go 能够在保持生产力的同时，大幅度提高性能。比如，全球知名的非营利教育组织可汗学院从 2019 年末开始，就将其在线教育平台的实现 [从 Python 迁移到了 Go](#)。虽然 Go 代码行数要多于 Python，但他们收获了近 10 倍的性能提升。

如果你立志或者已经上手云开发，那你就更应该马上开始学习 Go 语言。现在，Go 已经成为了云基础架构语言，它在云原生基础设施、中间件与云服务领域大放异彩。同时，GO 在 DevOps/SRE、区块链、命令行交互程序（CLI）、Web 服务，还有数据处理等方面也有大量拥趸，我们甚至可以看到 Go 在微控制器、机器人、游戏领域也有广泛应用。

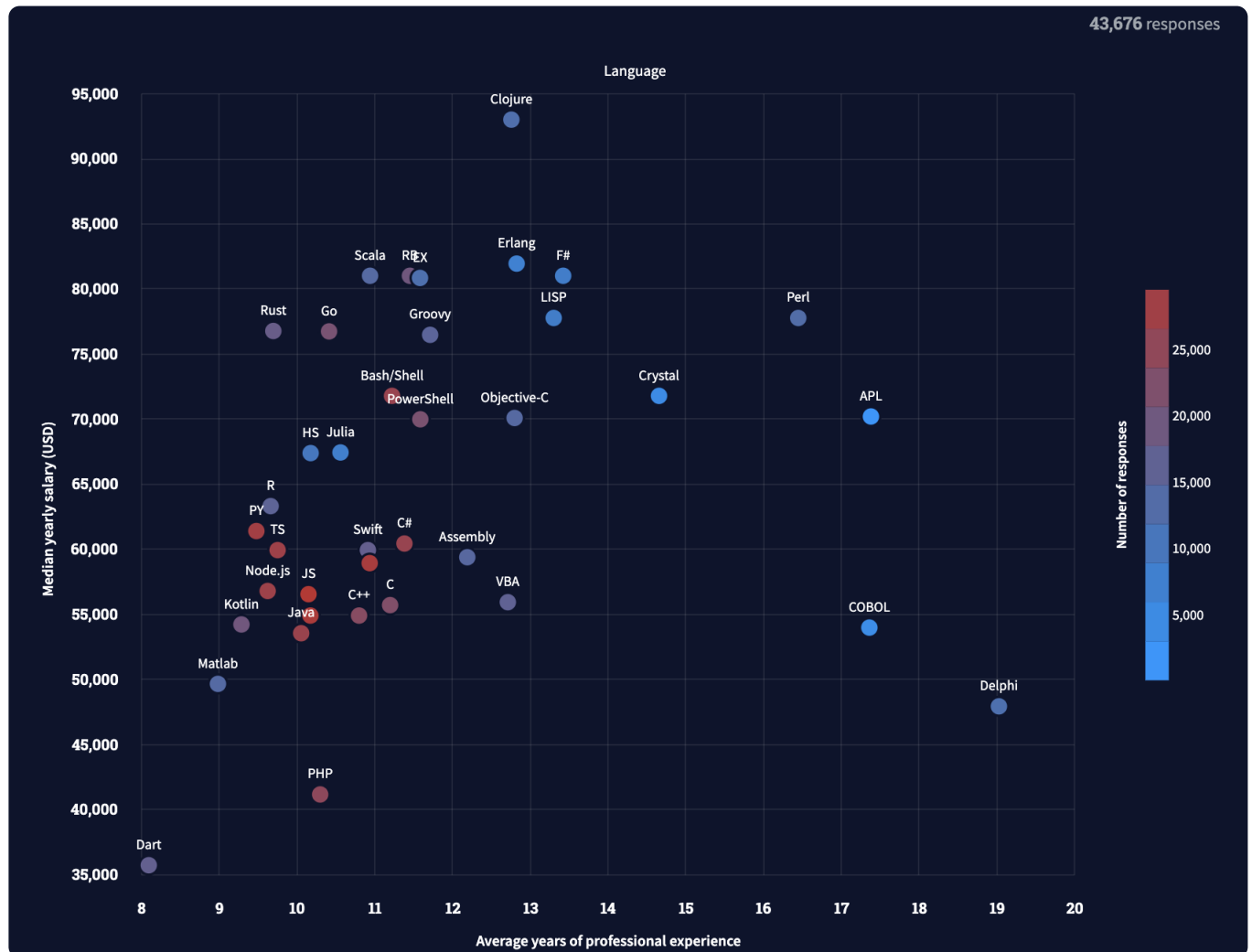
第三个理由：快乐又有“钱景”。

Go 最初的目标，就是重新为开发人员带来快乐。这个快乐来自哪里呢？相比 C/C++，甚至是 Java 来说，Go 在开发体验上的确有很大提升。笼统来说，这个提升来自于简单的语法、得心应手的工具链、丰富和健壮的标准库，还有生产力与性能的完美结合、免除内存管理的心智负担，对并发设计的原生支持等等。而这些良好的体验，恰恰是你写 Go 代码时的快乐源泉。

当然了，我相信你学习和使用 Go 肯定不是为了自嗨。运用 Go 体现自身价值，赢得理想职位才是最终目标。在十年后的今天，无论是在国内还是国外，无论是在大厂还是初创小公司，Go 都有着广泛的应用。Go 语言人才越来越抢手，对他们的争夺也日益激烈。

有报告表明，在腾讯、字节跳动、Uber 等许多公司，Go 都是极其受欢迎，在字节跳动、Uber 内部甚至已经成长为主力语言。

更何况，相对于 C、C++ 以及 Java 等主流语言，Go 语言人才目前仍处于蓝海阶段。根据 [stackoverflow 2021 调查报告](#) 的结果，仅考虑主流语言的话，Go 语言平均薪水位于头部位置。



这还仅仅是以欧美开发人员调查数据为主的计算结果。在 Go 更加火爆的国内，Go 的平均薪水水平位次可能还要更高。

怎样学才能少走弯路？

看到 Go 语言对初学者如此友好，又是生产力与性能结合得最好的语言，写起来还能体会到快乐。关键是当前国内外互联网大厂、初创小厂也都广泛接纳并应用 Go，就业“钱景”极佳！很多人都纷纷投身于 Go 语言的学习中，但是盲目的“一头热”只会让你多走许多弯路。

我总结了一下，最常见的无非就是这几个：

“入错行”，从开始到放弃。如果你在最开始缺乏对这门语言的认真评估，盲目投入进去，后期沉没的时间和精力成本都会巨大；

不动手。语言学习不是“纸上谈兵”，要动手去用，并且动手越早效果越好；

用其他编程语言思维编写 Go 代码。我认为，每门编程语言都有着自己独特的编程思维方式，如果你用其他编程语言的思维方式去写 Go 代码，那只能“形神皆丧”，无法掌握语言真谛；

没有建立起“设计意识”。编程语言学习的最终目的是写出具有现实实用意义的程序，所以你要培养自己选择适当的语言元素构造程序骨架的能力，也就是“设计意识”。尤其是要弄清楚不同语言元素所在的层次，不然你只能停留在“Hello, World”的世界里。

那么，我们到底要怎么学才能学好 Go 呢？

学好 Go 的前提是能坚持学下去。而要保证持续学下去，做好 Go 入门学习就至关重要。入门学习就好比一座在建大厦的地基，只有地基坚实、稳固，大厦才可能迎来建成，并耸立云霄的那天。

那么如何做好入门学习呢？这里告诉你**三个诀窍与五个阶段**。所谓三个诀窍是“**心定、手勤、脑勤**”。什么意思呢？我将这个入门课的学习分为下面五个阶段，我会结合这五个阶段来和你说明这三个诀窍。

第一个阶段：前置篇，“心定”建立认同感。

这一部分，我会带你了解 Go 的前世今生和设计哲学。如果你是有其他语言编程经验的开发人员，你就更应该完成前置篇的学习了。

这一部分存在的意义就是让你“心定”。所谓“心定”就是为了建立对 Go 语言的认同感。这种认同感是全方位的，包括对 Go 语言的设计目标、设计哲学、演化思路，还有社区行为规范等等。只有“心定”，才能避免出现“Hello-and-Bye”的情况，这是学好 Go 的前提。

第二个阶段：入门篇，“手勤”多动手实践。

在这一部分中，我将告诉你在不同平台上安装各种 Go 版本的方法，带你了解一个 Go 程序应该长什么模样，看看一些实用 Go 程序都有哪些语法元素和结构。关于 Go 的版本，如果我在课程中没有特地说明，那便默认使用的是 Go 最新的稳定版本，这里请你注意下。

编程不是“纸上谈兵”。我们最终都是要将编写完的源码提交给计算机编译运行的，因此，我希望你在这部分多动手、多实践。我在入门篇中将会让你拥有“照猫画虎”的能力。只有拥有这种能力，你才能“随心所欲”地动手实践。

第三个阶段：基础篇，“脑勤”多理解，夯实基础。

这一部分，我们会围绕着“程序 = 数据 + 算法”的逻辑，从变量、常量等基本概念，到数据类型，再到广义的算法，让你可以用 Go 建立对现实世界的抽象认知，也能明白 Go 程序运行的基本逻辑。

在基础篇的结尾，我们会结合已学习的基础语法做一个小练习项目。实践与理论的结合才能达到更好的学习效果。

第四个阶段：核心篇，“脑勤 +” 建立自己的 Go 应用设计意识。

在这部分，我会跟你介绍 Go 语言独有或经过较大创新的接口类型与 goroutine 等并发原语类型，这些语法元素是 Go 语言的核心。

Go 接口与 Goroutine 等并发原语类型有一个共同的特点，那就是它们都是可以影响到 Go 应用程序的结构设计的语法元素。Goroutine 等并发原语是 Go 应用运行时并发设计的基础，而接口则是 Go 推崇的面向组合设计的抓手，这一动一静共同构成了 Go 应用程序的骨架。通过这一部分，你就能建立自己的 Go 应用“设计意识”。

第五个阶段：实战篇，攻克 Go 开发的“最后一公里”。

编程就是要做到学以致用。在掌握了 Go 语言的基础语法、核心语法并建立起自己的“设计意识”后，我们就是时候应用这些 Go 语言的特性来解决实际问题了。

在这部分中，我们将通过一个实战的例子，展示如何做好学习与使用之间的衔接，帮助你走完“使用 Go 进行生产级开发”这“最后一公里”。



更具体的目录，我也放在了这里，你可以看一下：

《Tony Bai · Go 语言第一课》课程目录

■ 开篇词 | 这样入门 Go，才能少走弯路

前置篇：心定之旅

01 前世今生：你不得不了解的 Go 的历史和现状

02 拒绝“Hello and Bye”：Go 语言的设计哲学是怎么一回事？

入门篇：勤加练手

03 配好环境：选择一种最适合你的 Go 安装方法

04 初窥门径：一个 Go 程序的结构是怎样的？

05 标准先行：Go 项目的布局标准是什么？

06 构建模式：Go 是怎么解决包依赖管理问题的？

07 构建模式：Go Modules 的 6 类常规操作

08 入口函数与包的初始化：搞清 Go 程序的执行次序

09 即学即练：构建一个 Web 服务就是这么简单

基础篇：“脑勤”多理解

10 变量声明：静态语言有别于动态语言的重要特征

11 代码块与作用域：如何保证变量不会被遮蔽？

12 基本数据类型：Go 原生支持的数值类型有哪些？

13 基本数据类型：为什么 Go 要原生支持字符串类型？

14 常量：Go 在“常量”设计上的创新有哪些？

15 同构复合类型：从定长数组到变长切片

16 复合数据类型：原生 Map 类型的实现机制是怎样的？

17 复合数据类型：用结构体建立对真实世界的抽象

18 控制结构：if 的“快乐路径”原则

19 控制结构：Go 的 for 循环，仅此一种

20 控制结构：Go 中的 switch 语句有哪些变化？

21 函数：请叫我“一等公民”

22 函数：怎么结合多返回值进行错误处理？

23 函数：怎么让函数更简洁健壮？

24 方法：理解“方法”的本质

25 方法：如何选择集合与 receiver 类型？

26 方法：怎么用类型嵌入模拟实现“继承”？

27 即学即练：跟踪函数调用链，理解代码更直观

加餐 | 他山之玉：学习 Go 你还可以参考什么？

核心篇：“脑勤 +” 洞彻核心

28 接口：接口即契约

29 接口：为什么 nil 接口不等于 nil？

30 接口：Go 中最强大的魔法

31 并发：轻量级线程 goroutine 是怎么回事？

32 并发：小 channel 中蕴含大智慧

33 并发：如何使用共享变量？

34 即学即练：如何实现一个轻量级线程池？

加餐 | 怎么拉取私有的 Go Module ?

实战篇：打通“最后一公里”

35 打稳根基：怎么实现一个 TCP 服务器？（上）

36 代码操练：怎么实现一个 TCP 服务器？（中）

37 成果检验：怎么实现一个 TCP 服务器？（下）

加餐 | 作为 Go Module 的作者，你应该知道的几件事

加餐 | 聊聊最近大热的 Go 泛型

期终测评周

■ 期末测评 | 来赴一场 Go 语言的满分之约

■ 结束语 | 和你一起迎接 Go 的黄金十年

在开篇词的最后，我想说，Go 是一门非常优秀的后端编程语言，它简单而不失表达力，兼具高生产力与战斗力（高性能）。它既能给你带去编码的快乐，也能因市场的广泛接受与热捧而提升你的个人价值。

所以，我衷心地希望你能完成这门课程的学习。希望我的这门课能帮助你将 Go 语言之路走得更顺畅，早日成长为一名优秀的 Go 语言开发工程师。

不要再犹豫了，来和我一起开启 Go 语言的学习之旅吧。

分享给需要的人，Ta 订阅后你可得 **20 元现金奖励**

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

下一篇 01 | 前世今生：你不得不了解的Go的历史和现状

精选留言 (27)

💬 写留言



自由

2021-10-13

Tony Bai 老师，你好，我接触 go 目前已经 3 个月了。在接触 go 一个月后，我就选择跳槽去了一家 go 的公司，我对 go 的发展是坚定不移的肯定，相信它会越来越好。

我的问题是，阅读完该专栏，我是否可以得到 go 风格的代码编写风格、优雅的 go 编程姿势？

展开 ∨

作者回复：感谢你对专栏的支持。

虽然这门课的定位是入门课，而并非进阶课，但我在课程讲解以及go示例代码中都会尽力以native的go代码去呈现。并且课程讲解穿插着一些关于go编码的最佳实践建议，希望你在阅读后能有收获。

btw，要写出native 的go代码，一定要多读高质量go代码，Go标准库是一个最好的选择。俗话说：“熟读唐诗三百首，不会作诗也会吟”，多读高质量代码，与此有异曲同工之妙。



👍 8



lesserror

2021-10-13

很早就关注老师的微博了，看到老师几乎每天都在分享关于Go的知识。

关于Go的入门课程看过很多了，目前认为普遍存在的问题有以下几个：要么过于侧重理论，脱离了实践；要么泛泛而谈，重点内容也没有说清楚；要么就是基于以前的gopath项目管理去讲解的；要么没有结合动态语言的特性来对比Go语言的不同，不能让动态语言...

展开 ▾

作者回复: 没有一门课是完美的，从你的问题来看，我只能说我的专栏能尽可能多的满足你的需求。

关于“一动一静”，“动”主要指程序的并发设计层面，如何设计去管理和控制goroutine。当程序运行起来后，真正“动”的是一个一个goroutine。而“静”，则是go源码中的实体以及它们之间的耦合关系。

1

4

**逍遥布顽童**

2021-10-13

终于等到课程上线，卡点刷新，第一时间购买学习。不知道自己是第一个，还是第二个购买的？

1

3

**夏周一**

2021-10-13

第一时间买下

展开 ▾

1

2



2021-10-13

收获了三个诀窍与五个阶段。所谓三个诀窍是“心定、手勤、脑勤”，大道至简，适用于任何语言。

1

2

**小韩**

2021-10-14

老师，什么样的人适合学 Go 语言，我目前是Java工程师，在一家小公司里。

作者回复: Go设计之初，其目标是成为一门通用的系统编程语言。这一目标基本上就将go划分到后端编程语言行列。虽然go社区在前端、移动端编程的支持上面都做了很多尝试，比如：gopherjs项目以及go支持编译为webassembly来应对前端开发，再比如gomobile项目（<https://pkg.go.dev/golang.org/x/mobile>）让go也可以在移动端编程占有一席之地，但这么多年下来，go的主力战场还是云原生基础设施、中间件、web/api服务、微服务、命令行应用等等。因此如果你的目标与这些领域重合，那么go是一个很有竞争力的选择。



1

**aoe**

2021-10-14

学习Go是因为VS Code的Copilot插件支持Go但不支持Java，AI插件写代码不是一般的强

作者回复: Copilot插件我还没体验过，如果真的如果所言，那也是Go语言的一大幸事👍。

1

1

**罗杰**

2021-10-13

Tony 老师的粉丝来啦！

展开

作者回复: 感谢对我的专栏的支持👍。



1

**吉祥**

2021-10-13

从0开始学起来

展开

作者回复: 加油，努力学到最后！



1

**学了忘**

2021-10-13

已买，一定要坚持看下去啊！！！！

展开

作者回复: 加油！



1



2021-10-13

新鲜出炉，打个卡

展开 ∨



1

**陈东**

2021-10-15

跟进学习go，让go成为自己的生产力武器。谢谢您。

作者回复：希望go能早日成为你强大的生产力

**聪少 Jeff**

2021-10-15

好好学习，基础入门到提高📖

展开 ∨

作者回复：嗯嗯，有问题随时留言交流。

**华刚**

2021-10-15

开到课程，第一时间入手

展开 ∨

作者回复：感谢对课程的支持，希望课程能对你有所帮助。

**崆峒道**

2021-10-15

请问中小公司中的Go语言技术栈的岗位多吗？

展开 ∨

作者回复：go是生产力与执行效率两方面都有突出表现的语言。这两方面都能给中小公司省下不少money。一线城市接纳新语言的开发者较多，招聘也不再是问题了。因此我觉得一线城市应该

不少，这方面具体数据还得看招聘网站。二三线城市这些年go也在拓展地盘。在我地处的东北地区，越来越多小公司选用了go，趋势是好的。

**fomy**

2021-10-15

我是Java开发者，没有系统学习过Go语言，希望老师能说一下Java和Go的区别。

作者回复: 这是很大的话题，也是一个极容易“引战”的话题。

看待这个问题有多种维度，比如从语言语法、生产力、性能、社区活跃度，生态成熟度、发展前景等等。

语言语法见仁见智，java是不折不扣的面向对象编程语言，就像“java编程思想”一书中说：“一切都是对象”。而go是传统的命令式编程语言，按照go语言family图谱，它的先祖来自C、Pascal、Newsqueak等。语法简单，但谈不上“领先”，就像很多人说的在最近10年出品的编程语言中，go的语法显得有些“土气”，我更喜欢称之为朴实无华。很多人就像我，就是喜欢这种朴实。虽然朴实，但go的表达力并不差哦。

在生产力方面，目前来看go是要高于java的。

性能方面，同资源消耗下，go也是要高于java的。另外一点就是即便是新手写go，性能也不会很差。

社区活跃度方面，两者都是主流语言，java诞生年头多，且是目前企业应用领域的第一语言，其社区自然更好一些。生态成熟度也是如此，现在很难找到一个领域没有java的开源实现。实话说，go在这方面规模还不及java，但是增长速度要更快。

至于，发展前景，两者都是自己擅长领域的佼佼者，都有不错的前景。go由于处于成长期，蓝海属性更强一些。

**罗列**

2021-10-15

希望自己能学有所成，跳槽到更好的公司，给自己加油。

作者回复: 希望课程能给你带去帮助，成为一名更优秀的Gopher。

**SMTCode**

2021-10-14

老师在上海瑞驰还是沈阳瑞驰？

展开 ▾

作者回复：沈阳睿驰研发总部

**薛定谔猫的小确幸**

2021-10-14

非科班出身，工作几年了现在只会java一门语言，来学一下go，动静结合，哈哈！

作者回复：java还算不上“动”吧^_^。

**arch**

2021-10-14

tony老师，文中提及go GC很多次，期待补充些golang GC相关的文章哈

作者回复：对于任何一门带gc的语言，gc都是高级话题。对于入门第一课，gc不是重点。我这里后续会根据大家需求，考虑是否以加餐形式对gc做系统说明。大家有有关gc问题也可以随时在留言区提问。

